

Feature Similarity in using KNN for Text Segmentation based on Text Categorization

Taeho Jo
President
Alpha AI Research
Cheongju, South Korea
tjo018@naver.com

Abstract—In this research, we propose the KNN algorithm which considers the feature similarities, for automating the text segmentation by introducing the text categorization. In the previous work, we proposed the modernized version as an approach to the text segmentation, but must provide the domain tag, in addition to the text, manually. In this research, we modify the KNN algorithm into the version which considers the feature similarities and apply it to the text segmentation and the text classification. In the proposed system, only a text without its domain is given as the input, the domain is assigned to the text by the text classification, and each paragraph pair is classified into boundary or continuance by the classifier which corresponds to the domain. The goals of this research are to automate completely the text segmentation by combining it with the text classification and to improve the performance of both tasks, using the modernized KNN version.

I. INTRODUCTION

The text segmentation is the process of segmenting a text into subtexts based on its contents. The text segmentation is to decide whether a boundary is put between paragraphs, or not, so the task is converted into a binary classification of adjacent paragraph pairs into boundary or continuance. The process of text segmentation is to partition a text which is given as the input into paragraphs, generate adjacent paragraph pairs by sliding the two-sized window on the paragraphs, classify each of them into one of the two paragraphs. A boundary is put between paragraphs in a pair which is classified into boundary, and a text is segmented into subtexts, following the boundaries. Because the classification which is mapped from the text segmentation is a domain dependent task where a same entity is classified differently depending on the domain, it should be presented together with the text as the input.

This research is motivated by the requirement of knowing the domain in advance for executing the text segmentation. It is defined as an independent classification task domain by domain; a classifier is allocated to each domain independently of others. A domain is needed for nominating a corresponding classifier in segmenting a text. Before performing the text segmentation, the process of assigning a domain to a text is automated by the text classification. The text segmentation relates to the text classification for presenting only a text without its domain.

We adopt the KNN algorithm which considers the feature similarities for implementing the text segmentation which relates to the text classification. We define the similarity metric between numerical vectors which considers the feature similarities and modify the KNN algorithm using it. The modified version of KNN algorithm is applied to the text classification which assigns a domain automatically to an input text. A text is partitioned into paragraphs, and each of adjacent paragraph pairs is classified into boundary or continuance by the modified version. A boundary is put between paragraphs in each pair which is classified into boundary, and subtexts are extracted as independent texts.

Let us mention the benefits from this research. The poor discrimination among sparse vectors is solved by defining the similarity metric between numerical vectors which considers the feature similarities. We obtain the chance of modifying other machine learning algorithms using the proposed similarity metric. The performance of both the text segmentation and the text classification is improved by adopting the modified KNN algorithm for implementing both tasks. We automate the process of assigning a domain to a text for activating the text segmentation system by combining the text segmentation with the text classification.

Let us mention remaining tasks for reinforcing this research. The feature similarities may be computed with only some features rather than with all features for reducing the computation time. Paragraph pairs and texts may be encoded into compound representations, each of which consists of more than one structured form. Other machine learning algorithms or deep learning algorithms may be modified into proposed versions as approaches to both the text segmentation and the text classification. The text segmentation system which relates to the text classification may be implemented by adopting the proposed KNN version as a real program or a library.

II. PROPOSED WORKS

A. Encoding Process

This section is concerned with the process of encoding a text into a numerical vector. Words are used as features for encoding a text so, and some are selected from words in a corpus. The similarity between words is computed based on texts with their collocations and is used as the similarity

between features. The similarities among features are considered for computing the semantic similarity between texts. In this section, we describe the process of encoding a text into a numerical vector and the process of computing the similarity between texts.

The process of encoding texts into numerical vectors is illustrated in Figure 1. The entire text collection is indexed into a list of words which are feature candidates. Some words are selected as the features by criteria among the feature candidates in the second step. The weights of the selected features in the text are computed as their relationships with the text, as elements of the numerical vector. Refer to [1] for studying the process and the selection criteria in detail.

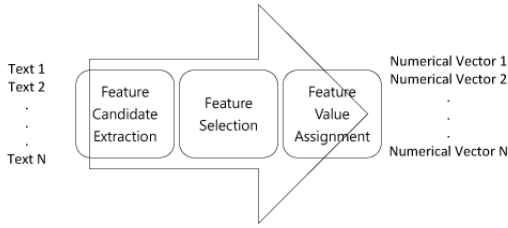


Figure 1. Process of Encoding Words into Numerical Vectors

The frame of computing the similarity between two numerical vectors is illustrated in Figure 2. The two d dimensional vectors and the d features are respectively notated respectively by $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_d]$, $\mathbf{y} = [y_1 \ y_2 \ \dots \ y_d]$, and $f_1, f_2, \dots, f_d$. The feature similarity refers to the similarity between two features, which is notated by $\text{sim}(f_i, f_j)$. The feature value similarity is the similarity between two vectors, $\mathbf{x}$ and $\mathbf{y}$, such as cosine similarity. The similarity between words is computed by considering the two kinds of similarities.

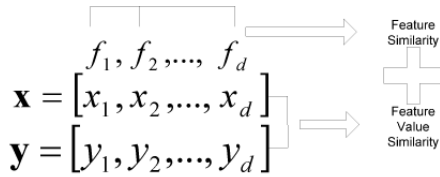


Figure 2. Frame of Computing Similarity between Numerical Vectors

Let us mention the process of computing the similarity between numerical vectors as the similarity between texts. The similarity between the features, f_i and f_j , is notated by $\text{sim}(f_i, f_j) = s_{ij}$. The similarity between the features is computed by equation (1),

$$s_{ij} = \text{sim}(t_i, t_j) = \frac{2 \times \#(t_i \wedge t_j)}{\#(t_i) + \#(t_j)}, \quad (1)$$

where $\#(t_i \wedge t_j)$, is the number of texts which include both words, t_i and t_j , $\#(t_i)$ is the number of texts which include

the word, t_i , and $\#(t_j)$ is the number of texts which include the word, t_j , and the similarity matrix to the d features is constructed as a $d \times d$ square matrix as follows:

$$S = \begin{pmatrix} s_{11} & s_{12} & \dots & s_{1d} \\ s_{21} & s_{22} & \dots & s_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ s_{d1} & s_{d2} & \dots & s_{dd} \end{pmatrix}.$$

The similarity between the two numerical vectors, $\mathbf{x}$ and $\mathbf{y}$, is computed by equation (2),

$$\text{sim}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^d \sum_{j=1}^d s_{ij} x_i y_j}{d \|\mathbf{x}\| \|\mathbf{y}\|} \quad (2)$$

Equation (2) is used for computing the similarity between a novice text and a sample text in the KNN algorithm.

Let us make some remarks on the encoding process and the computation of the similarity between two texts. A text is encoded into a numerical vector by the feature candidate extraction, the feature extraction, and the feature value assignment. The similarity between features which are given as words is computed by equation (1). The similarity between numerical vectors which represent texts is computed by equation (2). In future, we consider computing the similarities among some features rather than all features for improving the computation speed.

B. Feature Similarity based KNN Algorithm

This section is concerned with the KNN algorithm which considers the feature similarities. The version was initially proposed as the approach to the text classification by Jo in 2019 [2]. The similarity metric between numerical vectors which was described in the previous section is used for computing the similarity between a novice vector and a training example. The labels of its nearest neighbors are voted with their identical weights for decoding the label of a novice vector. This section is intended to describe the initial version of the KNN algorithm from which its variants are derived.

The process of computing the similarity between a novice item and a training example is illustrated in Figure 3. The labeled words which are gathered as samples are encoded into numerical vectors, and they are notated by a set $Tr = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$. A novice word is encoded into a numerical vector notated by $\mathbf{x}$. Its similarities with the numerical vectors which represent the sample words are computed by equation (2), as $\text{sim}(\mathbf{x}, \mathbf{x}_1), \text{sim}(\mathbf{x}, \mathbf{x}_2), \dots, \text{sim}(\mathbf{x}, \mathbf{x}_N)$. Some training examples which are most similar as the novice input are selected as its nearest neighbors.

In Figure 4, the process of selecting nearest neighbors by the ranking selection is illustrated. The training examples are sorted by the descending order of their similarities, after computing their similarities with a novice example. The

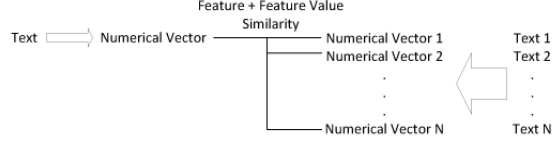


Figure 3. Similarities of Novice Input with Training Examples

training examples with their higher similarities with a novice example are selected as its nearest neighbors, as expressed in equation (3),

$$Nr = \text{Select}_k(Tr, \mathbf{x}), Nr \subseteq Tr \quad (3)$$

The set of nearest neighbors, Nr , is composed with elements as expressed in equation (4),

$$Nr = \{\mathbf{x}_1^{near}, \mathbf{x}_2^{near}, \dots, \mathbf{x}_k^{near}\} \quad (4)$$

The elements in the set, Nr , are used for voting their labels in the next step.

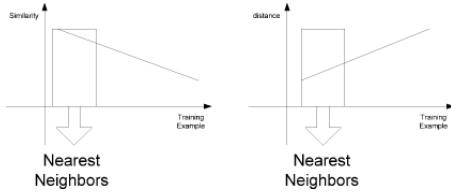


Figure 4. Selection of Most Similar or Least Distant Training Examples as Nearest Neighbors

The process of voting the labels of nearest neighbors for decoding the label of a novice word is illustrated in Figure 5. The predefined categories are notated by $c_1, c_2, \dots, c_m$, and the label of a nearest neighbor is notated by $c_j = \text{label}(\mathbf{x}_i^{near})$. The number of nearest neighbors which belong to the category, c_j , is notated by $\text{Count}(Nr, c_j)$. The label of the novice item, $\mathbf{x}$, is decided by equation (5),

$$\text{label}(\mathbf{x}) = \underset{j=1}{\text{argmax}}^m \text{Count}(Nr, c_j) \quad (5)$$

The process of deciding the label of a novice item by equation (5), is called voting.

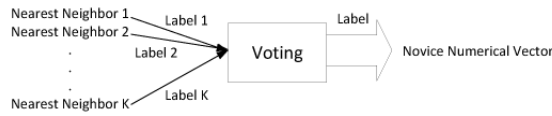


Figure 5. Voting of Labels of Nearest Neighbors for Deciding Label of Novice Input

Let us make some remarks on the KNN algorithm which considers the feature similarities. It computes the similarities of a novice item with the training examples. The training examples with their highest similarities with the novice item are selected as its nearest neighbors. The label of the novice

item is decided by voting the labels of its nearest neighbors. The ranking selection is adopted for selecting the nearest neighbors from training examples, in this version.

C. System Architecture

This section is concerned with the text segmentation system which adopts the KNN algorithm which considers the feature similarities. In Section II-B, we described the proposed version of KNN algorithm as the approach to the text segmentation. It is viewed as a binary classification which classifies a paragraph pair into boundary or continuance. This section is intended to describe the text segmentation system with respect to its function and its architecture.

The process of gathering sample paragraph pairs and classifying a novice one, is illustrated in Figure 6. Because even a same paragraph pair may be classified differently depending on the domain, the task is called domain dependent classification. Sample paragraph pairs are gathered domain by domain, and each pair is labeled with boundary or continuance. The paragraph pairs are taken from texts which are tagged with their same domain, each paragraph pair is classified into boundary or continuance. Sample paragraph pairs which are labeled with one of the two categories are encoded into numerical vectors in each domain.

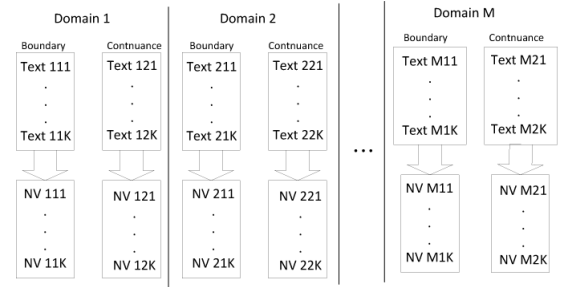


Figure 6. Preparation of Training Examples

The system architecture of the text segmentation system is illustrated in Figure 7. The text partition and sliding module partitions the input text into paragraphs and generate adjacent paragraph pairs by sliding the two sized window, and the encoding module encodes the adjacent paragraph pairs into numerical vectors by the process which is described in Section II-A. The similarity computation module computes the similarities of each paragraph pair with the samples which are labeled with boundary or continuance by the process which is described in Section II-B and select samples with their highest similarities as the nearest neighbors. The voting module votes the labels of the nearest neighbors for deciding the label of the novice one. In this system, the adjacent paragraphs are classified into boundary or continuance, and a boundary is put between paragraphs in each pair which is classified into boundary.

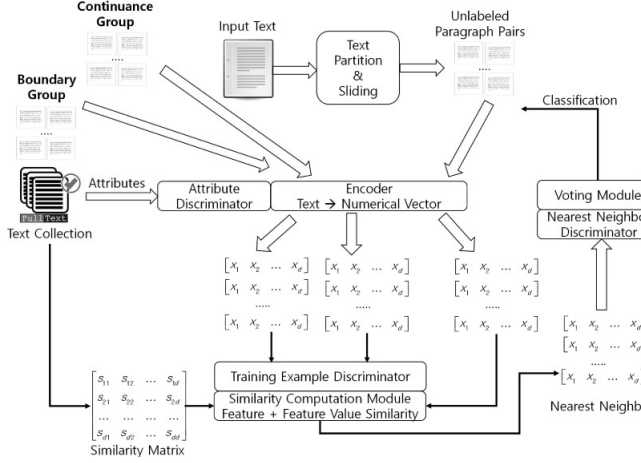


Figure 7. System Architecture

The execution flow of the text segmentation system is illustrated in Figure 8. Texts with same domain are initially collected, adjacent paragraph pairs are extracted from them, and the paragraph pairs are labeled with boundary or continuance. From an input text, adjacent paragraph pairs are extracted, and are encoded into numerical vectors, together with the sample paragraph pairs. The adjacent paragraph pairs are classified into boundary or continuance, and there are two groups of paragraph pairs: boundary group and continuance group. The boundary is marked between paragraphs in each pair in the boundary group, and text is partitioned by the marked boundaries into subtexts as the final output of this system.

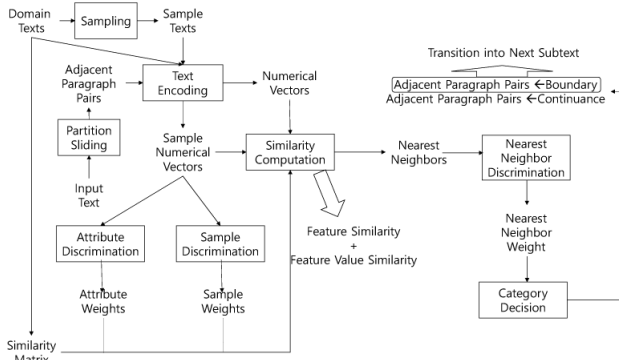


Figure 8. System Flow

Let us make some remarks on the system architecture and the execution flow of the text segmentation system. In this research, the text segmentation is viewed as the binary classification of adjacent paragraph pairs and the similarity metric between numerical vectors is proposed. The KNN algorithm is modified by the similarity metric, and the version is adopted for implementing the text segmentation system. The system architecture and the execution flow are

presented; this indicates that this research stay in the step of making the general design of the system. In next research, we try to make the detail design and the implementation of the entire text segmentation system.

D. Connection with Text Classification

This section is concerned with the connection of the text segmentation with the text classification. In the previous section, we mentioned the text segmentation system as an instance of domain dependent classification. The domain is automatically decided for nominating a classifier by connecting the keyword with the text classification. The KNN algorithm which was described in Section II-B is used for implementing the text classification. This section is intended to describe the connection of the text segmentation system with the text classification for automating the decision of the domain.

The system architecture of the text categorization system which consists of the encoding module, the similarity computation module, and the voting module is illustrated in Figure 9. The encoding module is for encoding texts into numerical vectors by the process which is described in Section II-A. The similarity computation module is for computing the similarities between numerical vectors which represent a novice text and a sample text and selecting ones with their highest similarities as the nearest neighbors. The voting module is for deciding the label of the novice item by voting the labels of the nearest neighbors. This system used for automating deciding the domain of the input text.

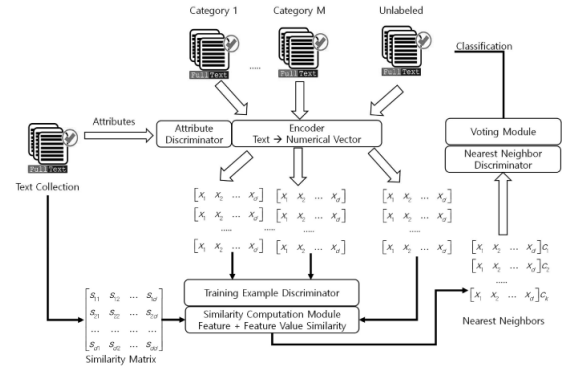


Figure 9. Text Categorization System

The connection of the text segmentation with the text classification is illustrated in Figure 10. The texts each of which is labeled with its own domain are prepared as the sample texts, and the approach to the text classification is trained with them. A domain is assigned to a novice text by classifying it into a domain automatically. The novice text is provided with its domain to the text segmentation system which is described in Section II-C. The role of the text classification system is to nominate the classifier

which corresponds to the domain for the text segmentation, automatically.

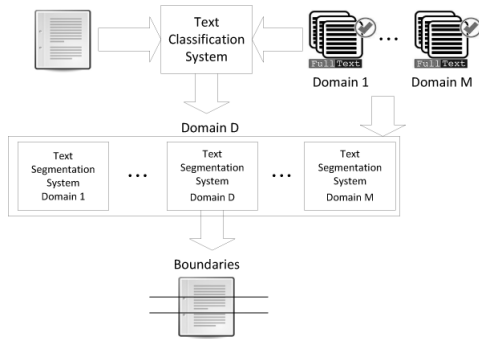


Figure 10. Text Segmentation System connected with Text categorization

The process of executing the text segmentation, connecting it with the text classification is illustrated in Figure 11. The sample texts each of which is labeled with its own domain, and sample paragraph pairs each of which is labeled with boundary and continuance are prepared in each domain. The novice text is classified into one among the predefined domains, and the classifier which correspond to the domain is nominated. Adjacent paragraph pairs are generated by sliding the two sized window on the paragraph list, and each paragraph pair is classified into boundary or continuance. Subtexts which are generated by segmenting the text following the boundaries are generated as the final output; the domain which is assigned to the input text is the guide for selecting a classifier.

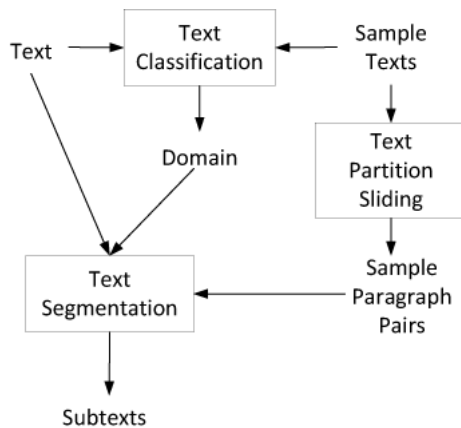


Figure 11. System Flow of Text Segmentation connected with Text Categorization

Let us make some remarks on the connection of the text segmentation with the text classification. The text classification is applied to the automatic decision of the input text domain. The role of text classification is to decide the domain for nominating a classifier, and the role of the text segmentation is to partition the entire text into subtexts

based on its content. In the execution flow, the input text is classified into a domain, paragraph pairs are generated by sliding the two sized window, and each paragraph pair is classified into boundary or continuance. We consider connecting the text classification as the main task the text segmentation as the opposite to what is proposed in this research.

REFERENCES

- [1] T. Jo, "Text Mining: Concepts and Big Data Challenge", Springer, 2019.
- [2] T. Jo, "Text Classification using Feature Similarity based K Nearest Neighbor", 13-21, AS Medical Science, 3, 2019.
- [3] T. Jo, "Content based Segmentation of News Articles using Feature Similarity based K Nearest Neighbor", 61-64, The Proceedings of 19st International Conference on Information and Knowledge Engineering, 2019.